UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/788,490	03/01/2004	Viera Bibr	T8468041US	9028

26912 7590 12/29/2009
GOWLING LAFLEUR HENDERSON LLP
SUITE 1600, 1 FIRST CANADIAN PLACE
100 KING STREET WEST
TORONTO, ON M5X 1G5
CANADA

EXAMINER	
WANG, BEN C	

ART UNIT	PAPER NUMBER
2192	

MAIL DATE	DELIVERY MODE
12/29/2009	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Office Action Summary	Application No.	Applicant(s)	
	10/788,490	BIBR ET AL.	
	Examiner	Art Unit	
	BEN C. WANG	2192	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE *3* MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *05 October 2009*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-36 and 38* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-36 and 38* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☒ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date *12/08/2009*.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .
5) ☐ Notice of Informal Patent Application
6) ☐ Other: _____.

DETAILED ACTION

1. A request for continued examination under 37 CFR 1.114, including the fee set forth in 37 CFR 1.17(e), was filed in this application after final rejection. Since this application is eligible for continued examination under 37 CFR 1.114, and the fee set forth in 37 CFR 1.17(e) has been timely paid, the finality of the previous Office action has been withdrawn pursuant to 37 CFR 1.114. Applicant's submission filed on October 5, 2009 has been entered.

2. Applicant's amendment dated October 5, 2009, responding to the Final Office action mailed May 11, 2009 provided in the rejection of claims 1-36 and 38; wherein claims 15, 18, 34, and 38 have been amended

Claims 1-36 and 38 remain pending in the application and which have been fully considered by the examiner.

Applicant's arguments with respect to claims currently amended have been fully considered but are moot in view of the new grounds of rejection – see *Warila et al.* - art made of record, as applied hereto.

Information Disclosure Statement

The information disclosure statement (IDS) submitted on December 8, 2009 was filed after the mailing date of the Office action on May 11, 2009. The submission is in compliance with the provisions of 37 CFR 1.97. Accordingly, the information disclosure statement is being considered by the examiner.

Claim Rejections – 35 USC § 103(a)

The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

3. Claims 1-12, 14, 18-29, 31, 36, and 38 are rejected under 35 U.S.C. 103(a) as being unpatentable over Hulai et al. (Pub. No. US 2003/0060896 A9) (hereinafter 'Hulai') in view of Warila et al. (Pub. No. US 2008/0313282 A1) (hereinafter 'Warila' - art made of record) and further in view of Thomas J. Carroll JR., (Pub. No. US 2002/0085020 A1) (hereinafter 'Carroll. Jr.')

4. **As to claim 1** (Previously Presented), Hulai discloses a method for generating a screen element, based on a data object, of a component application executing on a wireless device for display on a user interface of a wireless device, the component application including a data component having at least one data field definition and a screen component having at least one screen element definition, the components being defined in a structured definition language, the method comprising the steps of:

- selecting the screen component corresponding to the screen element selected for display (e.g., Fig. 1, element 18 – User Interface; Fig. 2, element 67 – screen generation engine; Fig. 4, element 48 – User

Interface Definition Section; [0031], Lines 5-8; [0035], Lines 1-3; [0049], Lines 1-7; Fig. 8, element of S802 – i.e., create screen object; Fig. 9; Figs. 12-14; [0112], Lines 1-11; [0113], Lines 1-4);

- selecting the data component mapped by the mapping according to the mapping identifier (e.g., Fig. 4, element 52 – Device Local Data Definition Section;[0105], Lines 6-9; [0122], Lines 1-7; [0049], Lines 9-11 – a local data definition section defining the format of data to be stored locally on the mobile device by the application);
- obtaining a data object field value corresponding to the data field definition of the mapped data component (e.g., Fig. 16I, Sec. 3.2.3.3; [0039], Lines 1-7 – each of object classes includes attributes used to store parameters defined by the XML file, and functions allowing the XML entity to be processed at the mobile device); and
- generating a screen element from the screen element definition according to the format of the data field definition as defined in the mapped data component (e.g., Fig. 4, element 48 – User Interface Definition Section; [0049], Lines 1-7; Fig. 5, elements 48, 54, 56, 58 – User Interface – Device; [0078]-[0079]; Fig. 8; [0091], Lines 6-11; [0095], Lines 1-6; [0098]; Fig. 12; [0114])

Further, Hulai discloses providing the device an application definition file, containing definitions for a user interface format for the application at the wireless device (e.g., Abstract) but does not explicitly disclose other limitations stated below.

However, in an analogous art of *User Interface, Operating System and Architecture*, Warila discloses generating a screen element from the screen element definition *to include the data object field value* (e.g., In light of the specification, which stated '... *all changes to the data component can be immediately reflected on the screen and vice versa* (e.g., Hulai teaches the user's input data mapping/updating (*device-to-server direction*) from the screen component (*device*) to data component (*server*)*; this model allows building effective wireless applications based on server-to-device notifications; the data updates asynchronously pushed from the server are instantaneously reflected at the UI screen*.' (Abstract), *Warila* discloses '… a defined portion of the application can be addressed and updated in response to application events without necessitating update of the entire application, the appearance and behavior of the application can be propagated with consistency across heterogeneous devices …' ([0032]), '… each editable area of the screen contains a value attribute that is updated automatically when the user operates the interface and changes information inside controls. these value attributes live within the superstructure itself …' ([0251]), [please also see Figure 4, elements 406 - Applications Data, 410 - Client Device, 404 - Server and Figure 18 – Current Value] - emphasis added)

Therefore, it would have been obvious to one of ordinary skill in the art, at the time the invention was made to combine the teachings of Warila into the Hulai's system to further provide other limitations stated above in the Hulai system.

The motivation is that it would further enhance the Hulai's system by taking, advancing and/or incorporating the Warila's system which offers significant advantages that the SimpleOS network model is that an application's code can be updated on the fly if so desired by the carrier or the developer as once suggested by Warila. (e.g., [0534)

Furthermore, Warila teaches a novel user interface, operating system, software language and architecture (e.g., Abstract) but Hulai and Warila do not explicitly disclose other limitations stated below.

However, in an analogous art of *XML-Based Graphical User Interface Application Development Tool Kit*, Carroll, Jr. discloses identifying at least one mapping present in the screen component, the mapping for specifying a relationship between the screen component and the data component as defined by an identifier representing the mapping (e.g., [0299] - … Actions are used to tie an application's user interface (i.e., the screen component) to an application's logic (i.e., the data component) … An application can then register to be notified when these actions or events occur, and can define what particular application logic can be executed when an action occurs …; [0300] - … a property file 110 can be used to map to a subset of graphical elements within the user interface definition file 104 …; [0218] – A hash table that maps the names of the element's attributes to their values; [0219] - … The configure method takes the hash table passed to the create method, and uses the values with this hash table to set the corresponding properties …; Fig. 24, elements 102 – Application Source Code; 104 – User Interface Definition (XML File); 110 – Property File; 112 – Parser; and

114 – Application GUI Interface (Run-Time); [0188] - … During the application compilation process, the parser 112 uses information both in the application source code 102 and the user interface definition file 104 (the data component) to create the applications' graphical user interface 114 (the screen component) …; [0017] - … to segregate the various application or software logic, and interface design aspect, of the application development process …; Fig. 22; [0270] - … to easily manipulate and specify, through a series of hierarchical properties and input fields (the data component), a particular graphical user interface definition. The data entered through this graphical-based tool can then be used to define or generate the source interface XML file, and hence the interface itself … A property file 110 can be used to map to a subset of graphical elements in the interface class library …; [0215] – [0218]; [0300]; P. 89, Left-Col., Lines 39-42 - … said application interface file is an extended markup language file which maps application screen component function calls to Java screen component classes)

Therefore, it would have been obvious to one of ordinary skill in the art, at the time the invention was made to combine the teachings of Carroll, Jr. into the Hulai-Warila's system to further provide other limitations stated above in the Hulai system.

The motivation is that it would further enhance the Hulai-Warila's system by taking, advancing and/or incorporating the Carroll, Jr.'s system which offers significant advantages that allow a developer to segregate the development of

the user interface from the development of the underlying application logic as once suggested by Carroll, Jr. (e.g., Abstract)

5. **As to claim 2** (Original) (incorporating the rejection in claim 1), Hulai discloses the method and the system wherein a plurality of the data field definitions of the data component is shared between the screen component and the data component as represented by the mapping (e.g., Fig. 16G, Sec. 2.2 – the key to ARML usage is the application definition file held on the AIRIX server. This file defines the AIRIX tables for the application, the allowed message set and the user interface definitions for the application on a given device)

6. **As to claim 3** (Original) (incorporating the rejection in claim 2), Hulai discloses the method further comprising the step of linking the plurality of data field definitions to corresponding ones of the screen element definitions of the screen component as represented by the identifier (e.g., Fig. 16S, Sec. 5.1.3.1 (The SCREEN tag) – an identifier for the screen; [0086]; [0097], Lines 6-8; [0098]; [0109])

7. **As to claim 4** (Original) (incorporating the rejection in claim 2), Hulai discloses the method further comprising the step of detecting a user event of the user interface related to the screen element (e.g., Fig. 9, elements of S918, S920, S922, S924; [0096], Lines 1-13; Fig. 10; [0101]-[0103])

8. **As to claim 5** (Original) (incorporating the rejection in claim 4), Hulai discloses the method further comprising the step of identifying the mapping in the screen component corresponding to the linked data component of the affected screen element (e.g., [0085]; [0086] – the particular identity of the mobile device on which the application is to be presented may be identified by a suitable identifier, in the form of a header contained in the server side application output; [0097] – virtual machine software further maintains a list identifying each instance of each event and action object, and an associated identifier of an event; i.e., Fig. 16II, Sec. 6.6.3.2, Sec. 6.6.3.3, Sec. 6.6.3.4; Fig. 16JJ, Sec. 6.7.3.2)

9. **As to claim 6** (Original) (incorporating the rejection in claim 5), Hulai discloses the method further comprising the step of updating the data object in a memory using the data field definition of the linked data component (e.g., Fig. 16K, Sec. 3.3.2, 3.3.3.3; Fig. 16R; Figs. 15A-15C; Fig. 16M, Sec. 3.3.4, Lines 1-3, Figure 4 – a sample package definition)

10. **As to claim 7** (Original) (incorporating the rejection in claim 5), Hulai discloses the method further comprising the step of creating a new one of the data object in a memory using the data field definition of the linked data component (e.g., Fig. 2; [0035]-[0036] – object classes corresponding to XML entities supported by the virtual machine software, and possibly contained within an application definition file)

11. **As to claim 8** (Previously Presented) (incorporating the rejection in claim 2, Hulai discloses the method and the system wherein the data object field value is obtained by being passed to the user interface as a screen parameter (e.g., [0039], Lines 1-7 – object classes define objects that allow device to process each of the supported XML entities at the mobile device; [0041], Lines 5-7 – at run time, instances of object classes corresponding to these classes are created and populated with parameters contained within application definition file, as required; i.e., Fig. 16L, Sec. 3.3.3.5)

12. **As to claim 9** (Original) (incorporating the rejection in claim 2), Hulai discloses the method and the system wherein a first screen element definition is mapped by a first one of the identifiers to a first one of the data components and a second screen element definition is mapped by a second one of the identifiers to a second one of the data components different from the first data component (e.g., [0085]; [0086] – the particular identity of the mobile device on which the application is to be presented may be identified by a suitable identifier, in the form of a header contained in the server side application output; [0097] – virtual machine software further maintains a list identifying each instance of each event and action object, and an associated identifier of an event; i.e., Fig. 16II, Sec. 6.6.3.2, Sec. 6.6.3.3, Sec. 6.6.3.4; Fig. 16JJ, Sec. 6.7.3.2)

13. **As to claim 10** (Original) (incorporating the rejection in claim 9), Hulai discloses the method and the system wherein the first screen element definition

and the second screen element definition are mapped to the same data component using the first identifier (e.g., [0085]; [0086] – the particular identity of the mobile device on which the application is to be presented may be identified by a suitable identifier, in the form of a header contained in the server side application output; [0097] – virtual machine software further maintains a list identifying each instance of each event and action object, and an associated identifier of an event; i.e., Fig. 16II, Sec. 6.6.3.2, Sec. 6.6.3.3, Sec. 6.6.3.4; Fig. 16JJ, Sec. 6.7.3.2)

14. **As to claims 11** (Original) (incorporating the rejection in claim 2), and Hulai discloses the method and the system wherein the structured definition language is XML based (e.g., Abstract, Lines 12-17)

15. **As to claim 12** (Original) (incorporating the rejection in claim 2), Hulai discloses the method and the system wherein the identifier is a simple primary key (e.g., [0070]; i.e., Fig. 15A, PK=LNGRECIPIENTID; Fig. 15B – primary key; Fig. 16I, Sec. 3.2.3.1 – PK – which of the table fields is the primary key for the table; Fig. 16J, Figure 2 – sample email schema, primary key, Figure 3 – a sample table definition section, PK=LNGMESSAGEID, PK=LNGRECIPIENTID)

16. **As to claim 14** (Original) (incorporating the rejection in claim 2), Hulai discloses the method further comprising the step of receiving an asynchronous communication message by the device via a network coupled to the device, the

message including a message data object (e.g., Fig. 1; Fig. 3; [0043]; Abstract, Lines 3-17; [0008] through [0011])

17. **As to claim 18** (Currently Amended), Hulai discloses a system for generating a screen element, based on a data object, of a component application executing on a wireless device, for display on a user interface of the wireless device, the component application including a data component having at least one data field definition and a screen component having at least one screen element definition, the component being defined in a structured definition language, the system having memory for storing computer readable instructions and a processor configured to execute the instructions, the instructions for providing:

- a data manager for obtaining a data object field value corresponding to the data field definition of the mapped data component (e.g., Fig. 16I, Sec. 3.2.3.3; [0039], Lines 1-7 – each of object classes includes attributes used to store parameters defined by the XML file, and functions allowing the XML entity to be processed at the mobile device); and
- a presentation manager for generating a screen element from the screen element definition according to the format of the data field definition as defined in the mapped data component (e.g., Fig. 4, element 48 – User Interface Definition Section; [0049], Lines 1-7; Fig. 5, elements 48, 54, 56, 58 – User Interface – Device; [0078]-[0079]; Fig. 8; [0091], Lines 6-11; [0095], Lines 1-6; [0098]; Fig. 12; [0114])

Further, Hulai discloses providing the device an application definition file, containing definitions for a user interface format for the application at the wireless device (e.g., Abstract) but does not explicitly disclose other limitations stated below.

However, in an analogous art of *User Interface, Operating System and Architecture*, Warila discloses for generating a screen element from the screen element definition to include the data object field value (e.g., In light of the specification, which stated '... *all changes to the data component can be immediately reflected on the screen and vice versa* (e.g., Hulai teaches the user's input data mapping/updating (*device-to-server direction*) from the screen component (*device*) to data component (*server*)*; this model allows building effective wireless applications based on server-to-device notifications; the data updates asynchronously pushed from the server are instantaneously reflected at the UI screen*.' (Abstract), *Warila* discloses '… a defined portion of the application can be addressed and updated in response to application events without necessitating update of the entire application, the appearance and behavior of the application can be propagated with consistency across heterogeneous devices …' ([0032]), '… each editable area of the screen contains a value attribute that is updated automatically when the user operates the interface and changes information inside controls. these value attributes live within the superstructure itself …' ([0251]), [please also see Figure 4, elements 406 - Applications Data, 410 - Client Device, 404 - Server and Figure 18 – Current Value] - emphasis added)

Therefore, it would have been obvious to one of ordinary skill in the art, at the time the invention was made to combine the teachings of Warila into the Hulai's system to further provide other limitations stated above in the Hulai system.

The motivation is that it would further enhance the Hulai's system by taking, advancing and/or incorporating the Warila's system which offers significant advantages that the SimpleOS network model is that an application's code can be updated on the fly if so desired by the carrier or the developer as once suggested by Warila. (e.g., [0534)

Furthermore, Warila teaches a novel user interface, operating system, software language and architecture (e.g., Abstract) but Hulai and Warila do not explicitly disclose other limitations stated below.

However, in an analogous art of *XML-Based Graphical User Interface Application Development Tool Kit*, Carroll, Jr. discloses a mapping manager for identifying at least one mapping present in the screen component, the mapping for specifying a relationship between the screen component and the data component as defined by an identifier representing the mapping, and for selecting the data component mapped by the mapping according to the mapping identifier (e.g., [0299] - … Actions are used to tie an application's user interface (i.e., the screen component) to an application's logic (i.e., the data component) … An application can then register to be notified when these actions or events occur, and can define what particular application logic can be executed when an action occurs …; [0300] - … a property file 110 can be used to map to a subset

of graphical elements within the user interface definition file 104 …; [0218] – A hash table that maps the names of the element's attributes to their values; [0219] - … The configure method takes the hash table passed to the create method, and uses the values with this hash table to set the corresponding properties …; Fig. 24, elements 102 – Application Source Code; 104 – User Interface Definition (XML File); 110 – Property File; 112 – Parser; and 114 – Application GUI Interface (Run-Time); [0188] - … During the application compilation process, the parser 112 uses information both in the application source code 102 and the user interface definition file 104 (the data component) to create the applications' graphical user interface 114 (the screen component) …; [0017] - … to segregate the various application or software logic, and interface design aspect, of the application development process …; Fig. 22; [0270 - … to easily manipulate and specify, through a series of hierarchical properties and input fields (the data component), a particular graphical user interface definition. The data entered through this graphical-based tool can then be used to define or generate the source interface XML file, and hence the interface itself … A property file 110 can be used to map to a subset of graphical elements in the interface class library …; [0215] – [0218]; [0300]; P. 89, Left-Col., Lines 39-42 - … said application interface file is an extended markup language file which maps application screen component function calls to Java screen component classes)

Therefore, it would have been obvious to one of ordinary skill in the art, at the time the invention was made to combine the teachings of Carroll, Jr. into the

Hulai-Warila's system to further provide other limitations stated above in the Hulai system.

The motivation is that it would further enhance the Hulai-Warila's system by taking, advancing and/or incorporating the Carroll, Jr.'s system which offers significant advantages that allow a developer to segregate the development of the user interface from the development of the underlying application logic as once suggested by Carroll, Jr. (e.g., Abstract)

18. **As to claim 19** (Original) (incorporating the rejection in claim 18), please refer to claim **13** as set forth accordingly.

19. **As to claim 20** (Original) (incorporating the rejection in claim 19), Hulai discloses the system wherein the plurality of data field definitions are linked to corresponding ones of the screen element definitions of the screen component as represented by the identifier (e.g., Fig. 16S, Sec. 5.1.3.1 (The SCREEN tag) – an identifier for the screen; [0086]; [0097], Lines 6-8; [0098]; [0109])

20. **As to claim 21** (Previously Presented) (incorporating the rejection in claim 19), Hulai discloses the system is further comprising the presentation manager configured for detecting a user event of the user interface related to the screen element (e.g., Fig. 9, elements of S918, S920, S922, S924; [0096], Lines 1-13; Fig. 10; [0101]-[0103])

21. **As to claim 22** (Previously Presented) (incorporating the rejection in claim 21), Hulai discloses the system further comprising the mapping manager is further configured for identifying the mapping in the screen component corresponding to the linked data component of the affected screen element (e.g., [0085]; [0086] – the particular identity of the mobile device on which the application is to be presented may be identified by a suitable identifier, in the form of a header contained in the server side application output; [0097] – virtual machine software further maintains a list identifying each instance of each event and action object, and an associated identifier of an event; i.e., Fig. 16II, Sec. 6.6.3.2, Sec. 6.6.3.3, Sec. 6.6.3.4; Fig. 16JJ, Sec. 6.7.3.2)

22. **As to claim 23** (Previously Presented) (incorporating the rejection in claim 22), Hulai discloses the system wherein the data manager is further configured for updating the data object in a memory using the data field definition of the linked data component (e.g., Fig. 16K, Sec. 3.3.2, 3.3.3.3; Fig. 16R; Figs. 15A-15C; Fig. 16M, Sec. 3.3.4, Lines 1-3, Figure 4 – a sample package definition)

23. **As to claim 24** (Previously Presented) (incorporating the rejection in claim 22), Hulai discloses the system wherein the data manager is further configured for creating a new one of the data object in a memory using the data field definition of the linked data component (e.g., Fig. 2; [0035]-[0036] – object classes corresponding to XML entities supported by the virtual machine software, and possibly contained within an application definition file)

24. **As to claim 25** (Previously Presented) (incorporating the rejection in claim 19), please refer to claim **8** as set forth accordingly.

25. **As to claim 26** (Original) (incorporating the rejection in claim 19), please refer to claim **9** as set forth accordingly.

26. **As to claim 27** (Original) (incorporating the rejection in claim 26), please refer to claim **10** as set forth accordingly.

27. **As to claim 28** (Previously Presented) (incorporating the rejection in claim 19), please refer to claim **11** as set forth accordingly.

28. **As to claim 29** (Original) (incorporating the rejection in claim 19), please refer to claim **12** as set forth accordingly.

29. **As to claim 31** (Original) (incorporating the rejection in claim 19), Hulai discloses the system further comprising a communication manager for receiving an asynchronous communication message by the device via a network coupled to the device, the message including a message data object (e.g., Fig. 1; Fig. 3; [0043]; Abstract, Lines 3-17; [0008] through [0011])

30. **As to claim 36** (Previously Presented), Hulai discloses a wireless device for generating a screen element, based on a data object, of a component application executing on the wireless device for display on a user interface of the wireless device, the component application including a data component having at least one data field definition and a screen component having at least one screen element definition, the component being defined in a structured definition language, the wireless device comprising the steps of:

- means for selecting the screen component corresponding to the screen element selected for display (e.g., Fig. 1, element 18 – User Interface; Fig. 2, element 67 – screen generation engine; Fig. 4, element 48 – User Interface Definition Section; [0031], Lines 5-8; [0035], Lines 1-3; [0049], Lines 1-7; Fig. 8, element of S802 – i.e., create screen object; Fig. 9; Figs. 12-14; [0112], Lines 1-11; [0113], Lines 1-4; [0049], Lines 4-7 – a user interface definition section, specific to the user interface for the device);
- means for selecting the data component mapped by the mapping (e.g., Fig. 4, element 52 – Device Local Data Definition Section;[0105], Lines 6-9; [0122], Lines 1-7; [0049], Lines 9-11 – a local data definition section defining the format of data to be stored locally on the mobile device by the application);
- means for obtaining a data object field value corresponding to the data field definition of the mapped data component (e.g., Fig. 16I, Sec. 3.2.3.3; [0039], Lines 1-7 – each of object classes includes attributes used to store

parameters defined by the XML file, and functions allowing the XML entity to be processed at the mobile device);

- means for generating a screen element from the screen element definition according to the format of the data field definition as defined in the mapped data component (e.g., Fig. 4, element 48 – User Interface Definition Section; [0049], Lines 1-7; Fig. 5, elements 48, 54, 56, 58 – User Interface – Device; [0078]-[0079]; Fig. 8; [0091], Lines 6-11; [0095], Lines 1-6; [0098]; Fig. 12; [0114])

Further, Hulai discloses providing the device an application definition file, containing definitions for a user interface format for the application at the wireless device (e.g., Abstract) but does not explicitly disclose other limitations stated below.

However, in an analogous art of *User Interface, Operating System and Architecture*, Warila discloses means for generating a screen element from the screen element definition to include the data object field value (e.g., In light of the specification, which stated '... *all changes to the data component can be immediately reflected on the screen and vice versa* (e.g., Hulai teaches the user's input data mapping/updating (*device-to-server direction*) from the screen component (*device*) to data component (*server*)*; this model allows building effective wireless applications based on server-to-device notifications; the data updates asynchronously pushed from the server are instantaneously reflected at the UI screen*.' (Abstract), *Warila* discloses '… a defined portion of the application can be addressed and updated in response to application events without

necessitating update of the entire application, the appearance and behavior of the application can be propagated with consistency across heterogeneous devices …' ([0032]), '… each editable area of the screen contains a value attribute that is updated automatically when the user operates the interface and changes information inside controls. these value attributes live within the superstructure itself …' ([0251]), [please also see Figure 4, elements 406 - Applications Data, 410 - Client Device, 404 - Server and Figure 18 – Current Value] - emphasis added)

Therefore, it would have been obvious to one of ordinary skill in the art, at the time the invention was made to combine the teachings of Warila into the Hulai's system to further provide other limitations stated above in the Hulai system.

The motivation is that it would further enhance the Hulai's system by taking, advancing and/or incorporating the Warila's system which offers significant advantages that the SimpleOS network model is that an application's code can be updated on the fly if so desired by the carrier or the developer as once suggested by Warila. (e.g., [0534)

Furthermore, Warila teaches a novel user interface, operating system, software language and architecture (e.g., Abstract) but Hulai and Warila do not explicitly disclose other limitations stated below.

However, in an analogous art of *XML-Based Graphical User Interface Application Development Tool Kit*, Carroll, Jr. discloses means for identifying at least one mapping present in the screen component, the mapping for specifying

a relationship between the screen component and the data component (e.g., [0299] - … Actions are used to tie an application's user interface (i.e., the screen component) to an application's logic (i.e., the data component) … An application can then register to be notified when these actions or events occur, and can define what particular application logic can be executed when an action occurs …; [0300] - … a property file 110 can be used to map to a subset of graphical elements within the user interface definition file 104 …; [0218] – A hash table that maps the names of the element's attributes to their values; [0219] - … The configure method takes the hash table passed to the create method, and uses the values with this hash table to set the corresponding properties …; Fig. 24, elements 102 – Application Source Code; 104 – User Interface Definition (XML File); 110 – Property File; 112 – Parser; and 114 – Application GUI Interface (Run-Time); [0188] - … During the application compilation process, the parser 112 uses information both in the application source code 102 and the user interface definition file 104 (the data component) to create the applications' graphical user interface 114 (the screen component) …; [0017] - … to segregate the various application or software logic, and interface design aspect, of the application development process …; Fig. 22; [0270 - … to easily manipulate and specify, through a series of hierarchical properties and input fields (the data component), a particular graphical user interface definition. The data entered through this graphical-based tool can then be used to define or generate the source interface XML file, and hence the interface itself … A property file 110 can be used to map to a subset of graphical elements in the interface class library …;

[0215] – [0218]; [0300]; P. 89, Left-Col., Lines 39-42 - … said application interface file is an extended markup language file which maps application screen component function calls to Java screen component classes)

Therefore, it would have been obvious to one of ordinary skill in the art, at the time the invention was made to combine the teachings of Carroll, Jr. into the Hulai-Warila's system to further provide other limitations stated above in the Hulai system.

The motivation is that it would further enhance the Hulai-Warila's system by taking, advancing and/or incorporating the Carroll, Jr.'s system which offers significant advantages that allow a developer to segregate the development of the user interface from the development of the underlying application logic as once suggested by Carroll, Jr. (e.g., Abstract)

31. **As to claim 38** (Currently Amended), Hulai discloses a computer readable medium comprising instructions for generating a screen element, based on a data object, of a component application executing on a wireless device for display on a user interface of the wireless device, the component application including a data component having at least one data field definition and a screen component having at least one screen element definition, the components being defined in a structured definition language, the instructions, when implemented on a computing device, cause the computing device, cause the computing device to implement the steps of:

- selecting the screen component corresponding to the screen element selected for display (e.g., Fig. 1, element 18 – User Interface; Fig. 2, element 67 – screen generation engine; Fig. 4, element 48 – User Interface Definition Section; [0031], Lines 5-8; [0035], Lines 1-3; [0049], Lines 1-7; Fig. 8, element of S802 – i.e., create screen object; Fig. 9; Figs. 12-14; [0112], Lines 1-11; [0113], Lines 1-4);
- selecting the data component mapped by the mapping according to the mapping identifier (e.g., Fig. 4, element 52 – Device Local Data Definition Section;[0105], Lines 6-9; [0122], Lines 1-7; [0049], Lines 9-11 – a local data definition section defining the format of data to be stored locally on the mobile device by the application);
- obtaining a data object field value corresponding to the data field definition of the mapped data component (e.g., Fig. 16I, Sec. 3.2.3.3; [0039], Lines 1-7 – each of object classes includes attributes used to store parameters defined by the XML file, and functions allowing the XML entity to be processed at the mobile device); and
- generating a screen element from the screen element definition according to the format of the data field definition as defined in the mapped data component (e.g., Fig. 4, element 48 – User Interface Definition Section; [0049], Lines 1-7; Fig. 5, elements 48, 54, 56, 58 – User Interface – Device; [0078]-[0079]; Fig. 8; [0091], Lines 6-11; [0095], Lines 1-6; [0098]; Fig. 12; [0114])

Further, Hulai discloses providing the device an application definition file, containing definitions for a user interface format for the application at the wireless device (e.g., Abstract) but does not explicitly disclose other limitations stated below.

However, in an analogous art of *User Interface, Operating System and Architecture*, Warila discloses generating a screen element from the screen element definition to include the data object field value (e.g., In light of the specification, which stated '... *all changes to the data component can be immediately reflected on the screen and vice versa* (e.g., Hulai teaches the user's input data mapping/updating (*device-to-server direction*) from the screen component (*device*) to data component (*server*)*; this model allows building effective wireless applications based on server-to-device notifications; the data updates asynchronously pushed from the server are instantaneously reflected at the UI screen*.' (Abstract), *Warila* discloses '… a defined portion of the application can be addressed and updated in response to application events without necessitating update of the entire application, the appearance and behavior of the application can be propagated with consistency across heterogeneous devices …' ([0032]), '… each editable area of the screen contains a value attribute that is updated automatically when the user operates the interface and changes information inside controls. these value attributes live within the superstructure itself …' ([0251]), [please also see Figure 4, elements 406 - Applications Data, 410 - Client Device, 404 - Server and Figure 18 – Current Value] - emphasis added)

Therefore, it would have been obvious to one of ordinary skill in the art, at the time the invention was made to combine the teachings of Warila into the Hulai's system to further provide other limitations stated above in the Hulai system.

The motivation is that it would further enhance the Hulai's system by taking, advancing and/or incorporating the Warila's system which offers significant advantages that the SimpleOS network model is that an application's code can be updated on the fly if so desired by the carrier or the developer as once suggested by Warila. (e.g., [0534)

Furthermore, Warila teaches a novel user interface, operating system, software language and architecture (e.g., Abstract) but Hulai and Warila do not explicitly disclose other limitations stated below.

However, in an analogous art of *XML-Based Graphical User Interface Application Development Tool Kit*, Carroll, Jr. discloses identifying at least one mapping present in the screen component, the mapping for specifying a relationship between the screen component and the data component as defined by an identifier representing the mapping (e.g., [0299] - … Actions are used to tie an application's user interface (i.e., the screen component) to an application's logic (i.e., the data component) … An application can then register to be notified when these actions or events occur, and can define what particular application logic can be executed when an action occurs …; [0300] - … a property file 110 can be used to map to a subset of graphical elements within the user interface definition file 104 …; [0218] – A hash table that maps the names of the element's

attributes to their values; [0219] - … The configure method takes the hash table passed to the create method, and uses the values with this hash table to set the corresponding properties …; Fig. 24, elements 102 – Application Source Code; 104 – User Interface Definition (XML File); 110 – Property File; 112 – Parser; and 114 – Application GUI Interface (Run-Time); [0188] - … During the application compilation process, the parser 112 uses information both in the application source code 102 and the user interface definition file 104 (the data component) to create the applications' graphical user interface 114 (the screen component) …; [0017] - … to segregate the various application or software logic, and interface design aspect, of the application development process …; Fig. 22; [0270 - … to easily manipulate and specify, through a series of hierarchical properties and input fields (the data component), a particular graphical user interface definition. The data entered through this graphical-based tool can then be used to define or generate the source interface XML file, and hence the interface itself … A property file 110 can be used to map to a subset of graphical elements in the interface class library …; [0215] – [0218]; [0300]; P. 89, Left-Col., Lines 39-42 - … said application interface file is an extended markup language file which maps application screen component function calls to Java screen component classes)

Therefore, it would have been obvious to one of ordinary skill in the art, at the time the invention was made to combine the teachings of Carroll, Jr. into the Hulai-Warila's system to further provide other limitations stated above in the Hulai system.

The motivation is that it would further enhance the Hulai-Warila's system by taking, advancing and/or incorporating the Carroll, Jr.'s system which offers significant advantages that allow a developer to segregate the development of the user interface from the development of the underlying application logic as once suggested by Carroll, Jr. (e.g., Abstract)

32. Claim 35 is rejected under 35 U.S.C. 103(a) as being unpatentable over Hulai in view of Carroll. Jr.

33. **As to claim 35** (Previously Presented), Hulai discloses a method for generating a data object of a component application executing on a wireless device based on a change in a screen element displayed on a user interface of a wireless device, the component application including a data component having at least one data field definition and a screen component having at least one screen element definition, the component being defined in a structured definition language, the method comprising the steps of:

- selecting the screen component corresponding to the screen element (e.g., Fig. 1, element 18 – User Interface; Fig. 2, element 67 – screen generation engine; Fig. 4, element 48 – User Interface Definition Section; [0031], Lines 5-8; [0035], Lines 1-3; [0049], Lines 1-7; Fig. 8, element of S802 – i.e., create screen object; Fig. 9; Figs. 12-14; [0112], Lines 1-11; [0113], Lines 1-4);

- selecting the data component mapped by the mapping (e.g., Fig. 4, element 52 – Device Local Data Definition Section;[0105], Lines 6-9; [0122], Lines 1-7; [0049], Lines 9-11 – a local data definition section defining the format of data to be stored locally on the mobile device by the application);
- obtaining a changed value from the screen element corresponding to the mapped data component (e.g., [0036] – parser may convert each XML tag contained in the application definition file, and its associated data to tokens, for later processing; Fig. 9; [0096], Lines 1-13; [0117], Lines 6-18);
- assigning the changed value to a data field value of the data object according to the format of the data field definition as defined in the mapped data component (e.g., Fig. 16K, Sec. 3.3.2, Sec. 3.3.3.3; Fig. 16R; Figs. 15A-15C; Fig. 16M, Sec. 3.3.4, Lines 1-3, Figure 4 – a sample package definition)

Further Hulai discloses providing the device an application definition file, containing definitions for a user interface format for the application at the wireless device (e.g., Abstract) but does not explicitly disclose other limitations stated below.

However, in an analogous art of *XML-Based Graphical User Interface Application Development Tool Kit*, Carroll, Jr. discloses identifying at least one mapping present in the screen component, the mapping for specifying a relationship between the screen component and the data component (e.g., [0299] - … Actions are used to tie an application's user interface (i.e., the screen

component) to an application's logic (i.e., the data component) … An application can then register to be notified when these actions or events occur, and can define what particular application logic can be executed when an action occurs …; [0300] - … a property file 110 can be used to map to a subset of graphical elements within the user interface definition file 104 …; [0218] – A hash table that maps the names of the element's attributes to their values; [0219] - … The configure method takes the hash table passed to the create method, and uses the values with this hash table to set the corresponding properties …; Fig. 24, elements 102 – Application Source Code; 104 – User Interface Definition (XML File); 110 – Property File; 112 – Parser; and 114 – Application GUI Interface (Run-Time); [0188] - … During the application compilation process, the parser 112 uses information both in the application source code 102 and the user interface definition file 104 (the data component) to create the applications' graphical user interface 114 (the screen component) …; [0017] - … to segregate the various application or software logic, and interface design aspect, of the application development process …; Fig. 22; [0270 - … to easily manipulate and specify, through a series of hierarchical properties and input fields (the data component), a particular graphical user interface definition. The data entered through this graphical-based tool can then be used to define or generate the source interface XML file, and hence the interface itself … A property file 110 can be used to map to a subset of graphical elements in the interface class library …; [0215] – [0218]; [0300]; P. 89, Left-Col., Lines 39-42 - … said application

interface file is an extended markup language file which maps application screen component function calls to Java screen component classes)

Therefore, it would have been obvious to one of ordinary skill in the art, at the time the invention was made to combine the teachings of Carroll, Jr. into the Hulai's system to further provide other limitations stated above in the Hulai system.

The motivation is that it would further enhance the Hulai's system by taking, advancing and/or incorporating the Carroll, Jr.'s system which offers significant advantages that allow a developer to segregate the development of the user interface from the development of the underlying application logic as once suggested by Carroll, Jr. (e.g., Abstract)

34. Claims 15-17 and 32-34 are rejected under 35 U.S.C. 103(a) as being unpatentable over Hulai in view of Warila and Carroll, Jr. and further in view of Saulpaugh et al., (Pat. No. US 7,010,573 B1) (hereinafter 'Saulpaugh')

35. **As to claim 15** (Currently Amended) (incorporating the rejection in claim 14), Hulai discloses employing Virtual Machine and XML messaging technologies (e.g., Abstract, Lines 12-17), but Hulai, Warila and Carroll, Jr. do not explicitly disclose the limitations stated below.

However, in an art of *message gates using a shared transport in a distributed computing environment*, Saulpaugh discloses checking the asynchronous communication message for the mapping corresponding to the

data component of the application provisioned on the device (e.g., Col. 7, Lines 1-6 – the messages may be in a data representation language such as eXtensible Mark0up Languages (XML), 12-16 – each such message may be sent through a client message gate that may verify the correctness of the message)

Therefore, it would have been obvious to one of ordinary skill in the art, at the time the invention was made to combine the teachings of Saulpaugh into the Hulai- Warila-Carroll, Jr.'s system to further provide the limitations stated above in the Hulai-Warila Carroll, Jr.'s system.

The motivation is that it would further enhance the Hulai-Warila-Carroll, Jr.'s system by taking, advancing and/or incorporating the Saulpaugh's system which offers significant advantages for providing a simple way to connect various types of intelligent devices to allow for communication and sharing of resources while avoiding the interoperability and complex configuration problems existing in conventional networks as once suggested by Saulpaugh (e.g., Col. 2, Lines 3-7)

36. **As to claim 16** (Previously Presented) (incorporating the rejection in claim 15), Hulai discloses the method further comprising the step of updating the message data object corresponding to the message in a memory using the data field definition of the linked data component and then reflecting that data change in the screen element linked to the data object (e.g., [0085]; [0086] – the particular identity of the mobile device on which the application is to be presented may be identified by a suitable identifier, in the form of a header contained in the server side application output; [0097] – virtual machine software further maintains

a list identifying each instance of each event and action object, and an associated identifier of an event; i.e., Fig. 16II, Sec. 6.6.3.2, Sec. 6.6.3.3, Sec. 6.6.3.4; Fig. 16JJ, Sec. 6.7.3.2; Fig. 9; [0096], Lines 16-19).

37. **As to claim 17** (Original) (incorporating the rejection in claim 15), Hulai discloses the method further comprising the step of creating the data object corresponding to the message in a memory using the data field definition of the linked data component ([0040], Lines 4-9; [0041], Lines 5-7; i.e., [0051]; Fig. 9; [0096], Lines 16-19)

38. **As to claim 32** (Previously Presented) (incorporating the rejection in claim 19), Saulpaugh discloses the system further comprising the mapping manager configured for checking the message for the mapping corresponding to the data component of the application provisioned on the device (e.g., Col. 7, Lines 1-6 – the messages may be in a data representation language such as eXtensible Mark0up Languages (XML), 12-16 – each such message may be sent through a client message gate that may verify the correctness of the message)

39. **As to claim 33** (Previously Presented) (incorporating the rejection in claim 32), Hulai discloses the system further comprising the data manager configured for updating the message data object in a memory using the data field definition of the linked data component (e.g., [0085]; [0086] – the particular identity of the mobile device on which the application is to be presented may be identified by a

suitable identifier, in the form of a header contained in the server side application output; [0097] – virtual machine software further maintains a list identifying each instance of each event and action object, and an associated identifier of an event; i.e., Fig. 16II, Sec. 6.6.3.2, Sec. 6.6.3.3, Sec. 6.6.3.4; Fig. 16JJ, Sec. 6.7.3.2; Fig. 9; [0096], Lines 16-19)

40. **As to claim 34** (Currently Amended) (incorporating the rejection in claim 32), Hulai discloses the system further comprising the data manager configured for creating the data object corresponding to the message in a memory using the data field definition of the linked data component (e.g., [0040], Lines 4-9; [0041], Lines 5-7; i.e., [0051]; Fig. 9; [0096], Lines 16-19)

41. Claims 13 and 30 are rejected under 35 U.S.C. 103(a) as being unpatentable over Hulai in view of Warila and Carroll, Jr. and further in view of Greene et al., (Pat. No. US 6,868,441 B2) (hereinafter 'Greene')

42. **As to claims 13** (Original) (incorporating the rejection in claim 2), Hulai discloses employing Virtual Machine and XML messaging technologies (e.g., Abstract, Lines 12-17), but Hulai, Warila and Carroll, Jr. do not explicitly disclose the limitations stated below.

However, in an art of *method and system for implementing a global ecosystem of interrelated services*, Greene discloses the method and the system wherein the identifier is a composite key (e.g., Col. 69, Lines 1-10 – for example,

the PK for a given entity might be a string or an integer, or it might be a composite key having more than one component).

Therefore, it would have been obvious to one of ordinary skill in the art, at the time the invention was made to combine the teachings of Greene into the Hulai- Warila-Carroll, Jr.'s system to further provide the limitations stated above in the Hulai- Warila-Carroll, Jr.'s system.

The motivation is that it would further enhance the Hulai-Warila-Carroll, Jr.'s system by taking, advancing and/or incorporating the Greene's system which offers advantages for providing alternate, domain specific primary keys that can be used by the specific application, or by custom logic within the entity implementation, and checked for uniqueness by the central entity manager, using for example, a hashing or directory service as once suggested by Greene (e.g., Col. 69, Lines 1-10)

43. **As to claim 30** (Original) (incorporating the rejection in claim 19), please refer to claim **13** as set forth accordingly.

Conclusion

44. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Ben C. Wang whose telephone number is 571-270-1240. The examiner can normally be reached on Monday - Friday, 8:00 a.m. - 5:00 p.m., EST.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Tuan Q. Dam can be reached on 571-272-3695. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

/Ben C Wang/
Ben C. Wang
Examiner, Art Unit 2192

/Michael J. Yigdall/
Primary Examiner, Art Unit 2192